

On Reductions from Multi-Domain Noninterference to the Two-Level Case

Oliver Woizekowski and Ron van der Meyden

¹ Department of Computer Science, Kiel University
`oliver.woizekowski@email.uni-kiel.de`

² School of Computer Science and Engineering, UNSW Australia
`meyden@cse.unsw.edu.au`

Abstract. The literature on information flow security with respect to transitive policies has been concentrated largely on the case of policies with two security domains, High and Low, because of a presumption that more general policies can be reduced to this two-domain case. The details of the reduction have not been the subject of careful study, however. Many works in the literature use a reduction based on a quantification over “Low-down” partitionings of domains into those below and those not below a given domain in the information flow order. A few use “High-up” partitionings of domains into those above and those not above a given domain. Our paper argues that more general “cut” partitionings are also appropriate, and studies the relationships between the resulting multi-domain notions of security when the basic notion for the two-domain case to which we reduce is either Nondeducibility on Inputs or Generalized Noninterference. The Low-down reduction is shown to be weaker than the others, and while the High-up reduction is sometimes equivalent to the cut reduction, both it and the Low-down reduction may have an undesirable property of non-monotonicity with respect to a natural ordering on policies. These results suggest that the cut-based partitioning yields a more robust general approach for reduction to the two-domain case.

Keywords: Noninterference, nondeterminism, information flow, covert channels, policies

1 Introduction

Information flow security is concerned with finding, preventing and understanding the unwanted flow of information within a system implementation. One of its applications is the detection of covert channels, which might arise due to hard-to-foresee side-effects in the combination of smaller components, or even have been deliberately planted in the implementation by a rogue systems designer.

In order to reason about information flow, one needs to decompose the system into information *domains*. Domains are thought of as active components (users, processes, pieces of hardware, organisational units, etc.) and change the system state by performing actions. Domains may also make observations of the

system state. One way for information to flow from one domain to another is for the actions of the first to change the observations of the second. To describe the allowed flows of information in the system, one can specify for each pair of domains in which directions a flow of information is permissible. This specification is called a *policy* and usually represented as a directed graph: two examples are depicted in Figure 1. Policies are generally taken to be reflexive relations, since nothing can prevent a domain from obtaining information about itself. Moreover, they are often assumed to be transitive, (i.e., if $A \mapsto B$ and $B \mapsto C$ then we must also have $A \mapsto C$) since if B may obtain information about A , and B may pass this information to C , then there is nothing to prevent C receiving information about A .³



Fig. 1. The two-level policy $H \not\mapsto L$ and a transitive MLS-style policy.

Policy (a) in Figure 1, which we call $H \not\mapsto L$, is the simplest and most-studied case. Here we have two domains H and L , where H is thought to possess high and L low level clearance in the system, and information flow is permitted from L to H , but prohibited in the other direction. In practice, a larger set of domains is used to represent different security classifications, such as Unclassified (U), Confidential (C), Secret (S) and Top Secret (TS), and each security level may moreover be partitioned into compartments representing different types of information relevant to ‘need to know’ restrictions. This leads to policies such as the transitive policy whose Hasse diagram is depicted in Figure 1(b). Here the Confidential classification has two independent compartment domains (C_1, C_2), as does the Secret classification (S_1, S_2).

Informally, the statement $u \mapsto v$ can be read as “ u ’s behaviour may influence v ’s observations” or “ v may deduce something about u ’s behaviour”. A first formal definition for this intuition, called *noninterference* was given by Goguen and Meseguer [4], in the context of a *deterministic* automaton-based model. A generalization to nondeterministic systems is desirable so one can extend information flow analysis to, for example, the use of unreliable components, randomness or underspecification. Several works (e.g., [5,6,7,8,9,10,11]) extended the theory to nondeterministic systems and richer semantic models such as process algebras, resulting in a multitude of security definitions for several kinds of models, and with different intentions in mind.

Much of this subsequent literature has confined itself to the two-domain policy $H \not\mapsto L$, because there has been a view that more complex policies can be treated by reduction to this case. One obvious way to do so, that we may call

³ We confine our attention in this paper to the transitive case. Works that have investigated intransitive information flow theory include [1], [2] and [3].

the *pointwise* approach, is to apply a two-domain notion of noninterference for each pair of domains u, v in the policy with $u \not\rightsquigarrow v$. However, even in the case of deterministic systems, this can be shown to fail to detect situations where a domain may have disjunctive knowledge about a pair of other domains, neither of which may interfere with it individually (we present an example of this in Section 4). Goguen and Meseguer [4] already address this deficiency by what we may call a *setwise* approach, which requires that for each domain u , the set of domains v with $v \not\rightsquigarrow u$ does not collectively interfere with u .

However, while the setwise definition deals with what an individual domain may learn about a group of other domains, it does not deal with what groups may learn about individuals, or other groups. Subsequent work in the literature has taken this issue of *collusion* into account in reducing to the two-domain case. For example, a survey by Ryan [11] states:

It might seem that we have lost generality by assuming that the alphabet of the system is partitioned into High and Low. In fact we can deal with more general MLS-style policy with a lattice of classifications by a set of non-interference constraints corresponding to the various lattice points. For each lattice point l we define High to be the union of the interfaces of agents whose clearance dominates that of l . Low will be the complement, i.e., the union of the interfaces of all agents whose clearance does not dominate that of l . Notice also that we are assuming that we can clump all the high-level users together and similarly all the low-level users. There is nothing to stop all the low users from colluding. Similarly any high-level user potentially has access to the inputs of all other high users. We are thus again making a worst-case assumption.

We call the kind of groupings that Ryan describes *High-up coalitions*, and interpret his comments as the suggestion to extend existing, already understood security definitions for $H \not\rightsquigarrow L$ to the multi-domain case by generating multiple instances of $H \not\rightsquigarrow L$ formed from the policy in question using High-up coalitions. Ryan’s High-up approach is used in some works (e.g., [12]), but many others (e.g., [13,14,15,16]) use instead a dual notion of *Low-down coalitions*, where for some domain l , the group L is taken to be the set of domains u with $u \mapsto l$ and H is taken to be the complement of this set.

Yet other groupings exist that are neither High-up nor Low-down coalitions. For example, in Figure 1(b), the grouping $L = \{U, C_1, C_2\}$ and $H = \{S_1, S_2, TS\}$, corresponds to neither a High-up nor a Low-down coalition. It seems no less reasonable to consider L to be a colluding group that is seeking to obtain H level information. Note that this grouping is a *cut* in the sense that there is no $u \in H$ and $v \in L$ such that $u \mapsto v$. Since in such a cut, domains in L cannot individually obtain information about domains in H , it is reasonable to expect that they should not be able to get such information collectively. This motivates a reduction to the two-domain case that quantifies over all cuts.

Our contribution in this paper is to consider this range of alternative reductions from multi-domain policies to the two-domain case, and to develop an understanding of how these definitions are related and which are reasonable.

Reductions must start with an existing notion of security for the two-domain case. We work with two basic security definitions: Generalized Noninterference, which was introduced in [17], and Nondeducibility on Inputs, first presented in [5]. Our analysis shows that the relationships between the resulting notions of security are subtle, and the adequacy of a reduction approach may depend on the base notion for the two-domain policy. Amongst other results, we show that:

1. When the basic notion for the two-domain case is Generalized Noninterference, High-up coalitions yield a notion that is strictly stronger than the notion based on Low-down coalitions, which in turn is stronger than the pointwise generalization. For Nondeducibility on Inputs, however, High-up coalitions and Low-down coalitions give independent notions of security. Low-down coalitions imply the setwise definition in this case, but High-up coalitions imply only the weaker pointwise version.
2. For Generalized Noninterference, High-up coalitions are ‘complete’ in the sense of being equivalent to a reduction quantifying over all cuts. However, this completeness result does not hold for Nondeducibility on Inputs, where cuts yield a stronger notion of security.
3. Not all the resulting notions of security have an expected property of monotonicity with respect to a natural restrictiveness order on policies. (Security of a system should be preserved when one relaxes policy constraints.) In particular, High-up coalitions with respect to Nondeducibility on Inputs does not have this property, and Low-down coalitions do not have this property for either Generalized Noninterference or Nondeducibility on Inputs.

These conclusions indicate that while Ryan’s proposal to use High-up coalitions is sometimes adequate, a reduction that quantifies over the larger set of all cut coalitions seems to yield the most generally robust approach for reducing multi-domain policies to the two-domain case.

The structure of the paper is as follows. In Section 2, we introduce our model and show how systems and policies are described. Our reductions will use two basic security definitions for two-domain policies that are recalled and generalized to their obvious pointwise versions for the multi-domain case in Section 3. Section 4 gives some examples showing why the pointwise versions are still weaker than required, and it is necessary to consider reductions using groupings of domains. The range of reductions we consider are defined in Section 5. Our main results are stated in Section 6, full proofs of which are given in Section 7. Finally, we conclude and motivate further research in Section 8.

2 Background: Systems and Policy Model

Notational conventions. Sequences are represented as xyz , or $x \cdot y \cdot z$ if it helps readability. The set of finite sequences over a set A is denoted A^* , and the empty sequence is denoted ε . The length of α is written as $|\alpha|$. We write $\alpha(i)$ to denote the element with index i of a sequence α , where $i \in \mathbb{N}$, and the first element of α is $\alpha(0)$. We let $\text{last}(\alpha)$ be the last element of α if α is non-empty, and let it be

undefined if α is empty. If $X \subseteq A$ and $\alpha \in A^*$ then let $\alpha|_X$ be the subsequence of α with only elements from X retained. The set of total functions from A to B is denoted B^A .

Systems. We use an automaton-based model similar to the original Goguen-Meseguer one from [4]. A *system* is a structure $(S, A, O, D, \Delta, \text{obs}, \text{dom}, s_I)$ with S a set of *states*, A a finite set of *actions*, D a finite set of domains with at least two members, O a finite set of *observations* such that A and O are disjoint, $\Delta \subseteq S \times A \times S$ a (nondeterministic) transition relation, $\text{obs}: D \times S \rightarrow O$ an observation function, $\text{dom}: A \rightarrow D$ an assignment of actions to domains, and s_I the initial state. We write $\text{obs}_u(s)$ for $\text{obs}(u, s)$. The value $\text{obs}_u(s)$ represents the observation the domain u makes when the system is in state s . Observations can also be interpreted as outputs from the system. For an action a , the domain $\text{dom}(a)$ is the domain from which a originates. The relation Δ is called *deterministic* if for all $s, s', s'' \in S$, $a \in A$: if $(s, a, s') \in \Delta$ and $(s, a, s'') \in \Delta$ then $s' = s''$. We assume systems to be *input-enabled*, i.e. that for every $s \in S$ and $a \in A$ there is $s' \in S$ with $(s, a, s') \in \Delta$. The assumption of input-enabledness is made to guarantee that the domains' reasoning is based on their actions and observations only and cannot use system blocking behaviour as a source of information.

A *run* of a system is a sequence $s_0 a_1 s_1 \dots a_n s_n \in S(AS)^*$ such that for $i < n$, we have $(s_i, a_i, s_{i+1}) \in \Delta$. It is *initial* if $s_0 = s_I$. If not explicitly mentioned otherwise, we always assume initial runs. The set of initial runs of a system \mathcal{M} will be denoted $\text{Runs}(\mathcal{M})$. For a run r , the subsequence of actions of r is denoted $\text{act}(r)$ and the subsequence of actions performed by a domain u is denoted $\text{act}_u(r)$.

Notational and diagrammatic conventions for systems. If u is a domain and A the action set of a system, we write A_u for the set of actions a with $\text{dom}(a) = u$. Similarly, for X a set of domains we write A_X for the set of actions a with $\text{dom}(a) \in X$. Systems are depicted as directed graphs, where the vertices contain the state names. Domain observations are written near the vertices that represent the states. Edges are labelled with action names and represent transitions from one state to another. The initial state is marked with an arrow that points to it. Self-looping edges are omitted when possible to reduce clutter: thus, the lack of an edge labelled by action a from state s (as would be required by input-enabledness) implies the existence of edge (s, a, s) .

Modelling information by views. We will be interested in an asynchronous semantics for information, and capture asynchrony by treating sequences that differ only by stuttering observations as indistinguishable. This can also be described as no domain having access to a global clock. Intuitively, systems can be imagined as distributed and domains as representing network hosts. From this intuition it follows, for a given domain u , that local state changes within domains distinct from u that do not provide a new observation to u must not generate a copy of u 's current observation. To this end, we use an 'absorptive

concatenation' operator $\hat{\circ}$ on sequences. For all sequences α and $b_0 \dots b_n$ we let $\alpha \hat{\circ} \varepsilon = \alpha$ and

$$\alpha \hat{\circ} b_0 \dots b_n = \begin{cases} \alpha \hat{\circ} b_1 \dots b_n & \text{if } \alpha \neq \varepsilon \text{ and } \text{last}(\alpha) = b_0 \\ (\alpha \cdot b_0) \hat{\circ} b_1 \dots b_n & \text{otherwise.} \end{cases}$$

One can imagine $\alpha \hat{\circ} \beta$ as $\alpha \cdot \beta$ with stuttering at the point of connection removed. The information a domain acquires over the course of a run is modelled by the notion of *view*. Considering systems as networks suggests that, during a run, a domain can only directly see the actions performed by itself. This is reflected in our definition of view by eliminating actions performed by all other domains. For a domain u the operator $\text{view}_u: \text{Runs}(\mathcal{M}) \rightarrow (A \cup O)^*$ is defined inductively: for the base case $r = s_I$ let $\text{view}_u(r) = \text{obs}_u(s_I)$. For all $r \in \text{Runs}(\mathcal{M})$ of the form $r = r'as$, where $r' \in \text{Runs}(\mathcal{M})$, $a \in A$ and $s \in S$, let

$$\text{view}_u(r) = \begin{cases} \text{view}_u(r') \cdot a \cdot \text{obs}_u(s) & \text{if } \text{dom}(a) = u \\ \text{view}_u(r') \hat{\circ} \text{obs}_u(s) & \text{otherwise.} \end{cases}$$

An element $\text{view}_u(r)$ is called a u *view*. The set of all u views in system \mathcal{M} is denoted $\text{Views}_u(\mathcal{M})$.

For an example of a view, see the system in Figure 2 (recall that we elide self-loops) and consider the run $r = s_I a s_1 b s_2 b s_2 a s_3$; the domains are given by the set $\{A, B\}$, the domain assignment is given by $\text{dom}(a) = A$ and $\text{dom}(b) = B$, and the observations made by domain B are depicted near the state names. We have $\text{view}_B(r) = \perp b1b12$.

Note that B does not notice the first transition in r because we have $\text{obs}_B(s_I) = \text{obs}_B(s_1)$. Domain B does, however, learn about the last transition in r due to $\text{obs}_B(s_2) \neq \text{obs}_B(s_3)$. With the network analogy mentioned above, the last transition might model a communication from A to B .

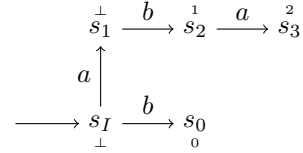


Fig. 2. System example.

Policies. A *policy* is a reflexive binary relation \mapsto over a set of domains D . We require \mapsto to

be reflexive because we assume that domains are aware of their own behaviour at all times. We assume also that policies are transitive, to avoid additional complexities associated with the semantics of intransitive policies. Transitive policies arise naturally from lattices of security levels. The policy that has received the most attention in the literature is over the set $D = \{H, L\}$, consisting of a domain H (or *High*), representing a high security domain whose activity needs to be protected, and a domain L (or *Low*), representing a low security attacker who aims to learn High secrets. We refer to this policy as $H \not\mapsto L$; it is given by the relation $\mapsto = \{(H, H), (L, L), (L, H)\}$.

If \mapsto is a policy over some domain set D , we write $u \mapsto$ for the set $\{v \in D : u \mapsto v\}$, and $\mapsto u$ for the set $\{v \in D : v \mapsto u\}$. Similarly, the expression $\not\mapsto u$ shall denote the set $\{v \in D : v \not\mapsto u\}$.

Further notational conventions for policies. Policies are depicted as directed graphs and their vertices carry domain names. Edges due to reflexivity or transitivity are omitted.

Policy abstractions and cuts. A set of domains can be abstracted by grouping its elements into sets. Such groupings can be motivated in a number of ways. One is simply that we wish to take a coarser view of the system, and reduce the number of domains by treating several domains as one. Groupings may also arise from several domains deciding to collude in an attack on the security of the system. Abstractions of a set of domains lead to associated abstractions of policies and systems.

An *abstraction* of a set of domains D is a set \mathcal{D} of subsets of D with $D = \bigcup_{F \in \mathcal{D}} F$ and $F \cap G \neq \emptyset$ implies $F = G$ for all $F, G \in \mathcal{D}$. Associated with each abstraction \mathcal{D} of D is a function $f_{\mathcal{D}}: D \rightarrow \mathcal{D}$ defined by taking $f_{\mathcal{D}}(u)$ to be the unique $F \in \mathcal{D}$ with $u \in F$. For a policy \mapsto over D we let $\mapsto^{\mathcal{D}}$ be the policy over \mathcal{D} defined by $F \mapsto^{\mathcal{D}} G$ if and only if there are $x \in F$ and $x' \in G$ with $x \mapsto x'$.

In order to formalize the idea of a reduction to $H \not\mapsto L$, we use abstractions that group all domains into two sets that correspond to the High and Low domains. A *cut* of a set of domains D with respect to a policy \mapsto is a tuple $\mathcal{C} = (\mathcal{H}, \mathcal{L})$ such that $\{\mathcal{H}, \mathcal{L}\}$ is an abstraction of D and there does not exist $u \in \mathcal{H}$ and $v \in \mathcal{L}$ with $u \mapsto v$. When forming policies, we identify cuts with their underlying abstractions, and write $\mapsto^{\mathcal{C}}$ for $\mapsto^{\{\mathcal{H}, \mathcal{L}\}}$, so the last requirement can also be formulated as $\mathcal{H} \not\mapsto^{\mathcal{C}} \mathcal{L}$. We mainly deal with abstractions that are given by cuts in this paper. See Figure 3 for an illustration of how policy (b) in Figure 1 is abstracted using $\mathcal{C} := (\mathcal{H}, \mathcal{L}) = (\{S_1, S_2, TS\}, \{U, C_1, C_2\})$, where we get $\mathcal{L} \mapsto^{\mathcal{C}} \mathcal{H}$ due to $C_1 \mapsto S_1$ or $C_2 \mapsto S_2$ and $\mathcal{H} \not\mapsto^{\mathcal{C}} \mathcal{L}$ as required for a cut.

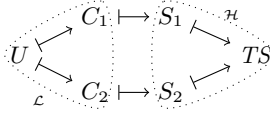


Fig. 3. Illustration of a policy abstraction.

Systems and abstractions. Systems can be viewed from the perspective of an abstraction. Intuitively, the actions of an abstract domain F are all the actions of any of its subdomains $u \in F$. It observes the collection of all observations made by the members of F and thus their observations are functions from F to O . Let $\mathcal{M} = (S, A, O, D, \Delta, \text{obs}, \text{dom}, s_I)$ be a system and \mathcal{D} be an abstraction of D . Then $\mathcal{M}^{\mathcal{D}}$ is the system $(S, A, O', \mathcal{D}, \Delta, \text{obs}^{\mathcal{D}}, \text{dom}^{\mathcal{D}}, s_I)$, where O' is the union of O^F for all $F \in \mathcal{D}$, its set of domains is \mathcal{D} , for a state $s \in S$, the observation $\text{obs}_F^{\mathcal{D}}(s)$ is the function with domain $F \in \mathcal{D}$ that sends each $x \in F$ to $\text{obs}_x(s)$, and $\text{dom}^{\mathcal{D}}(a) = f_{\mathcal{D}}(\text{dom}(a))$ for all $a \in A$. Intuitively, $\text{obs}_F^{\mathcal{D}}(s)$ records the observations made in each domain in F at s . Again, if $\mathcal{C} = (\mathcal{H}, \mathcal{L})$ is a cut we write $\mathcal{M}^{\mathcal{C}}$ for $\mathcal{M}^{\{\mathcal{H}, \mathcal{L}\}}$.

Monotonicity with respect to restrictiveness. In [18] the notion of *monotonicity with respect to restrictiveness* is discussed, which holds for a given notion of security X if, for all systems \mathcal{M} and policies \mapsto over the domain set of \mathcal{M} , the following statement holds: if \mathcal{M} is X -secure with respect to \mapsto then \mathcal{M} is X -secure with respect to every policy \mapsto' with $\mapsto \subseteq \mapsto'$. If a notion of security satisfies this property, we will say that it is *monotonic*. Intuitively, adding edges to a policy reduces the set of information flow restrictions $u \not\mapsto v$ implied by the policy, making the policy easier to satisfy, so one would expect every sensible notion of security to be monotonic. However, we will show that some notions of security obtained by a sensible construction based on cuts do not support this intuition.

3 Basic Notions of Noninterference

In this section we recall two security definitions which have been proposed in the literature for nondeterministic, asynchronous automaton-based models. We use these as the basic definitions of security for $H \not\mapsto L$ in the reductions that we study. For purposes of comparison, we state the definitions using the most obvious pointwise generalization from the usual two-domain case to the general multi-domain case.

For deterministic systems, we define an operator $\cdot : S \times A \rightarrow S$ by $s \cdot a = s'$, where s' is the unique state such that $(s, a, s') \in \Delta$ is satisfied. This operator is extended to action sequences by setting $s \cdot \varepsilon = s$ and $s \cdot (\alpha a) = (s \cdot \alpha) \cdot a$. This way action sequences ‘act’ on the system’s state set and $s \cdot \alpha$ is the state reached by performing α from s .

First, we recall Goguen and Meseguer’s original notion of noninterference from [19]. They introduced a function *purge*, which, for a given domain u , eliminates all actions that are supposed to be non-interfering with u . It can be inductively defined as follows: let $\text{purge}_u(\varepsilon) = \varepsilon$ and for all $\alpha \in A^*$ and $a \in A$, let

$$\text{purge}_u(\alpha a) = \begin{cases} \text{purge}_u(\alpha) \cdot a & \text{if } \text{dom}(a) \mapsto u \\ \text{purge}_u(\alpha) & \text{otherwise.} \end{cases}$$

Their idea is, with respect to a fixed domain u , to deem a system secure if all action sequences that are equivalent under purge_u yield the same observations for u .

Definition 1. *We say that \mathcal{M} is P-secure for \mapsto if for all $u \in D$ and $\alpha, \beta \in A^*$ we have: if $\text{purge}_u(\alpha) = \text{purge}_u(\beta)$ then $\text{obs}_u(s_I \cdot \alpha) = \text{obs}_u(s_I \cdot \beta)$.*

This statement can be understood as the requirement that observations made by a domain may only depend on the behaviour by domains permitted to interfere with it.

3.1 Nondeducibility on Inputs

Goguen and Meseguer’s definition of noninterference [19] was for deterministic systems only. Historically, Sutherland [5] was the first to consider information flow in nondeterministic systems. He presented a general scheme to instantiate notions of *Nondeducibility*, i.e., epistemic definitions of absence of information flows. The notion of Nondeducibility on Inputs is one instance of this general scheme.

Let $u, v \in D$. We say that $\alpha \in A_u^*$ and $\beta \in \text{Views}_v(\mathcal{M})$ are *v compatible* if there is $r \in \text{Runs}(\mathcal{M})$ with $\text{act}_u(r) = \alpha$ and $\text{view}_v(r) = \beta$. We write $u \rightsquigarrow_I v$ if there are $\alpha \in A_u^*$ and $\beta \in \text{Views}_v(\mathcal{M})$ which are not *v compatible*. In that case v gains information about u ’s behaviour in the following sense: if β is observed by v then v can deduce that u did not perform α . Nondeducibility $u \not\rightsquigarrow_I v$ therefore says that v is unable to make any nontrivial deductions about u behaviour. Applying this idea pointwise, we get the following definition of security:

Definition 2. A system is NDI_{pw} -secure for a policy \mapsto over domains D when for all $u, v \in D$: if $u \not\rightsquigarrow v$ then $u \not\rightsquigarrow_I v$.

In the case of the policy $H \not\rightsquigarrow L$ with just two domains, NDI_{pw} is the notion *Nondeducibility on Inputs* as it is usually defined. We denote it as just *NDI* in this case. The definition above generalizes this notion in one possible way to the multi-domain case. We discuss several others below.

3.2 Generalized Noninterference

The nondeducibility relation $H \not\rightsquigarrow L$ states that L considers all sequences of actions of H possible, but allows that L has some information about how these actions, if any, are interleaved with L ’s actions. See Figure 4 for a system that is *NDI*-secure but can be argued to leak information about how H ’s actions are interleaved into a run. The observations made by L are written near the state names.

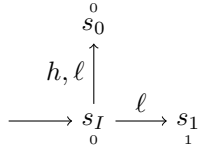


Fig. 4. System demonstrating a weakness of *NDI*.

This system is *NDI*-secure because every L view is compatible with every possible sequence of h actions performed by H . However, note that if the view $0l1$ is observed by L then it obtains the knowledge that it was the very first domain to act. The stronger notion of *Generalized Noninterference* introduced by McCullough [17] says that L does not have even this weaker

form of knowledge. The original formulation is for a two-domain policy and is based on a model that uses sets of event sequences. We present a straightforward multi-domain variant (that is similar to Mantel’s combination *BSI+BSD* [8]).

Definition 3. A system \mathcal{M} is GN_{pw} -secure for \mapsto if for all $u, v \in D$ with $u \not\rightsquigarrow v$, the properties

- $GN^+(u, v)$: for all $r \in \text{Runs}(\mathcal{M})$, for all $\alpha_0, \alpha_1 \in A^*$ with $\text{act}(r) = \alpha_0 \alpha_1$, and all $a \in A_u$ with there is $r' \in \text{Runs}(\mathcal{M})$ with $\text{act}(r') = \alpha_0 a \alpha_1$ and $\text{view}_v(r') = \text{view}_v(r)$, and
- $GN^-(u, v)$: for all $r \in \text{Runs}(\mathcal{M})$, all $\alpha_0, \alpha_1 \in A^*$ and all $a \in A_u$, with $\text{act}(r) = \alpha_0 a \alpha_1$, there is $r' \in \text{Runs}(\mathcal{M})$ with $\text{act}(r') = \alpha_0 \alpha_1$ and $\text{view}_v(r') = \text{view}_v(r)$.

are satisfied.

Intuitively, this definition says that actions of domains u with $u \nrightarrow v$ can be arbitrarily inserted and deleted, without changing the set of possible views that v can obtain. In the case of the two-domain policy $H \nrightarrow L$, the notion GN_{pw} is equivalent to the definition of Generalized Noninterference given in [20], and we denote this case by GN. Note that the system in Figure 4 is not GN-secure, because performing h as first action in a run makes it impossible for L to observe the view $0\ell 1$.

In *deterministic* systems, for the two-domain policy $H \nrightarrow L$, the notions NDI_{pw} and GN_{pw} , and Goguen and Meseguer's original notion of Noninterference are known to be equivalent. Thus, both NDI_{pw} and GN_{pw} are reasonable candidates for the generalization of Noninterference to nondeterministic systems.

4 Motivation for Abstraction

The definitions NDI_{pw} and GN_{pw} have generalized the corresponding definitions NDI and GN usually given for the two-domain policy $H \nrightarrow L$ in a *pointwise* fashion, stating in different ways that there should not be a flow of information from domain u to domain v when $u \nrightarrow v$. We now present some examples that suggest that these pointwise definitions may be weaker than required in the case of policies with more than two domains.

We first present an example which demonstrates that NDI_{pw} -security is flawed with respect to combined behaviour of multiple domains. (Interestingly, this can already be shown in a deterministic system.)

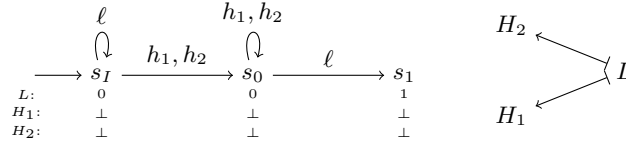


Fig. 5. A system and policy showing a weakness of NDI_{pw} .

Example 1. Consider the system and policy depicted in Figure 5. The domain assignment is given by $\text{dom}(l) = L$, $\text{dom}(h_1) = H_1$ and $\text{dom}(h_2) = H_2$. We have $H_1 \nrightarrow L$ and $H_2 \nrightarrow L$ and show that $H_1 \not\rightarrow_I L$ and $H_2 \not\rightarrow_I L$ hold. Let $\alpha = h_1^a$ for $a \geq 0$ and β be an L view, then β must have the form $0(\ell 0)^b(\ell 1)^c$, where $b, c \geq 0$. Consider the run $r = s_I(\ell s_I)^b h_2 s_0 (h_1 s_0)^a (\ell s_1)^c$, which satisfies $\text{view}_L(r) = 0(\ell 0)^b(\ell 1)^c = \beta$ and $\text{act}_{H_1}(r) = h_1^a$, and thus α and β are L compatible. Due to symmetry, we also get $H_2 \not\rightarrow_I L$ with the same argument.

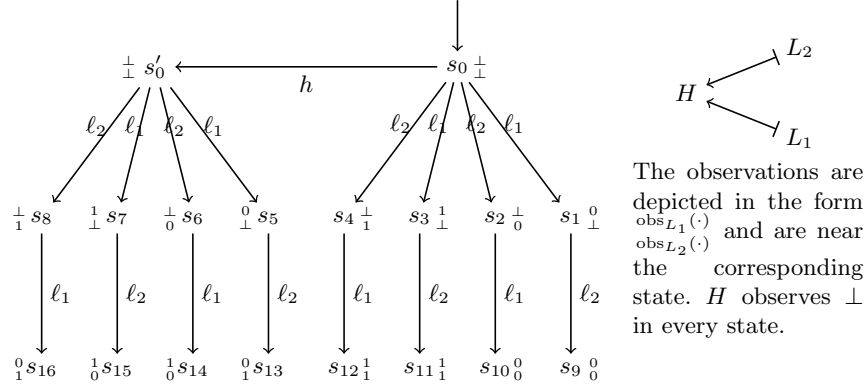


Fig. 6. System and policy illustrating a collusion attack.

The system therefore is NDI_{pw} -secure for the policy. However, if L observes the view $0\ell 1$ then H_1 or H_2 must have performed h_1 or h_2 , respectively. \square

In the example, domain L cannot know which of H_1 or H_2 was active upon observing the view $0\ell 1$, but L can tell that at least one of them was active nonetheless. It can be argued that this is a flow of information that is not permitted by the depicted policy. The example would turn formally insecure if we changed the policy to $H \not\vdash L$ and set $\text{dom}(h_1) = \text{dom}(h_2) = H$. The problem arises as soon as more than one domain must be noninterfering with L .

One way to address this weakness of NDI_{pw} is to revise the definition so that it deals with what a domain can learn about the actions of a set of domains collectively, rather than about these domains individually. We may extend the relation \rightsquigarrow_I to sets of domains as follows: for $X \subseteq D$, $X \neq \emptyset$ and $u \in D$, write $X \rightsquigarrow_I u$ if there are $\alpha \in A_X^*$ and $\beta \in \text{Views}_u(\mathcal{M})$ such that no $r \in \text{Runs}(\mathcal{M})$ satisfies both $\text{act}_X(r) = \alpha$ and $\text{view}_u(r) = \beta$. Applying this with the set $X = \not\vdash u$ consisting of all domains that may not interfere with domain u , we obtain the following setwise version of Nondeducibility on Inputs:

Definition 4. A system is NDI_{sw} -secure for \mapsto if for all $u \in D$, we have that $\not\vdash u \not\rightsquigarrow_I u$.

This gives a notion that is intermediate between the pointwise versions of Generalized Noninterference and Nondeducibility on Inputs:

Proposition 1. GN_{pw} is strictly contained in NDI_{sw} , and NDI_{sw} is strictly contained in NDI_{pw} . A system is NDI_{sw} -secure for $H \not\vdash L$ if and only if it is NDI_{pw} -secure for $H \not\vdash L$.

We remark that there is not a need to give a similar setwise definition of Generalized Noninterference, because the definition of GN_{pw} already allows the set of actions in a run to be modified, without change to the view of u , by arbitrary insertions and deletions of actions with domains v in $\not\vdash u$, through a sequence of applications of $\text{GN}^+(v, u)$ and $\text{GN}^-(v, u)$.

Despite NDI_{sw} and GN_{pw} being suitable for the multi-domain case and the latter notion being quite strict, one can argue that neither of them can handle collusion, where multiple domains join forces in order to attack the system as a team. The system depicted in Figure 6, a variant of Example 3 and Figure 4 from [21], can be shown to satisfy GN_{pw} -security, hence is secure in the strongest sense introduced so far. However, if L_1 and L_2 collude, they can infer from the parity of their observations that H performed h at the beginning of the run. This motivates the introduction of stronger *coalition-aware* notions of security.

5 Reduction-based Notions of Noninterference for Multi-domain Policies

The examples of the previous section indicate that in nondeterministic settings, it is necessary to deal with groups of agents both on the side of the attackers and the side of the domains being attacked. Policy cuts provide types of groupings and enable a reduction to a basic notion of security for two-domain policies. The question that then remains is what types of cut we should use, and which basic notion of security. In this section, we define three types of cut and the resulting notions of security when GN and NDI are taken to be the basic notion of security.

Let D be a set of domains. For $u \in D$ we define the following two special cuts $\text{Hu}(u)$ and $\text{Ld}(u)$.

$$\text{Hu}(u) := (u^{\mapsto}, D \setminus u^{\mapsto}) \text{ and } \text{Ld}(u) := (D \setminus \mapsto u, \mapsto u)$$

The term $\text{Hu}(u)$ stands for the cut that forms a High-up coalition starting at domain u , while $\text{Ld}(u)$ stands for the cut that forms a Low-down coalition with respect to u . Figure 7 depicts an example of each on the same policy.

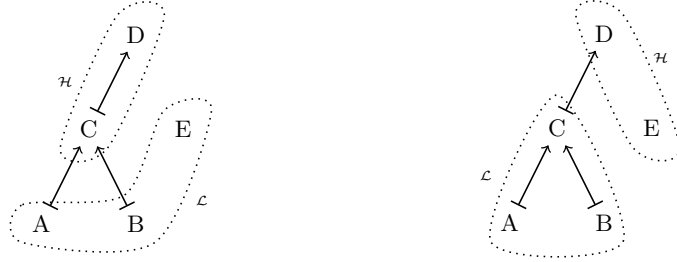


Fig. 7. Cuts $\text{Hu}(C)$ and $\text{Ld}(C)$ visualized.

Abstractions of type $\text{Hu}(\cdot)$ are suggested by Ryan (as discussed in the introduction), while the type $\text{Ld}(\cdot)$ is what we referred to as its dual. As already noted in the introduction, there are additional ‘cut’ abstractions that are neither High-up nor Low-down. In a systematic way, we can now obtain new notions of security based on cuts as follows.

Definition 5. Let \mathcal{M} be a system with domain set D and \mapsto be a policy over D . For $X \in \{GN, NDI\}$, we say \mathcal{M} is

- Cut X -secure (C - X -secure) for \mapsto , if \mathcal{M}^C is X -secure for \mapsto^C for all cuts C of D ,
- High-up X -secure (H - X -secure) for \mapsto , if $\mathcal{M}^{\text{Hu}(u)}$ is X -secure for $\mapsto^{\text{Hu}(u)}$ for all $u \in D$,
- Low-down X -secure (L - X -secure) for \mapsto , if $\mathcal{M}^{\text{Ld}(u)}$ is X -secure for $\mapsto^{\text{Ld}(u)}$ for all $u \in D$.

There is a straightforward relationship between these notions of GN and their NDI-counterparts.

Proposition 2. For all $X \in \{C, H, L\}$: the notion X -GN is strictly contained in X -NDI.

This follows directly from Definition 5, the fact that GN implies NDI due to Proposition 1, and that the system depicted in Figure 4 provides separation for each case. Also, one would expect that reasonable extensions of GN and NDI agree if applied to $H \not\vdash L$, and this is exactly what we find, since we can identify singleton coalitions with their only member.

6 Main Result

We now state the main result of the paper. We have a set of definitions of security that address the need to consider groupings of attackers and defenders in multi-domain policies, based on two basic notions of security NDI and GN for the two-domain case. We are now interested in understanding the relationships between these definitions. Additionally, we are interested in understanding which definitions satisfy the desirable property of monotonicity.

Theorem 1. The notions of GN_{pw} , L -GN, H -GN, C -GN, NDI_{pw} , NDI_{sw} , L -NDI, H -NDI and C -NDI-security are ordered by implication as depicted in Figure 8. The containment relations are strict; arrows due to reflexivity or transitivity are omitted. The name of a notion is underlined if and only if it is monotonic.

In particular, we find for the GN-variants that Ryan’s proposal to use reductions based on High-up coalitions is complete, in the sense that it yields the same notion of security as a quantification over all cuts. This notion is moreover adequate in the sense of being monotonic. Somewhat surprisingly, the dual notion based on Low-down coalitions is strictly weaker, and also fails to be monotonic.

The situation is different for the basic notion of NDI. In this case, we see that Ryan’s proposal is not complete with respect to quantification over all cuts. Indeed, the resulting notion H-NDI does not even imply the more adequate setwise version of NDI, although it does imply the pointwise version. The Low-down version of NDI does imply the setwise version, and is independent of H-NDI. However, neither H-NDI nor L-NDI is monotonic. This leaves the (monotonic) cut based variant as the most satisfactory notion in this case.

How one arrives at these results is explained in the next section, which gives proofs for all results in this paper.

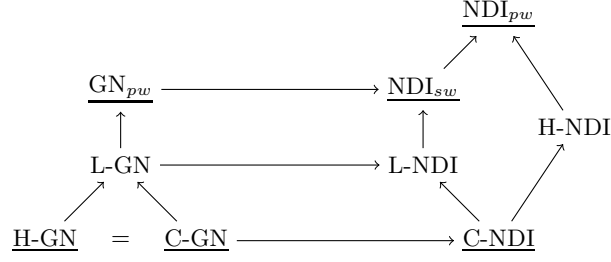


Fig. 8. Implications between our notions of security.

7 Technical Details

7.1 Relationship between GN_{pw} and NDI_{sw}

Proof (of Proposition 1). GN_{pw} is strictly contained in NDI_{sw} : For containment, assume a system to be GN_{pw} -secure for a policy \mapsto , let u be a domain and X be a set of domains in \mathcal{M} , where $x \not\mapsto u$ for all $x \in X$. Then the conditions $GN^+(x, u)$ and $GN^-(x, u)$ guarantee that at any position in the run, any action from A_X can be inserted or removed ‘without changing the u view of the run’. More precisely, one always finds a run with the same u view and the desired action inserted or removed at any position. Therefore, all u views are compatible with all sequences from A_X^* and the system is NDI_{pw} -secure for \mapsto . Separation is due to the system depicted in Figure 4 and the fact that NDI_{sw} and NDI are equivalent on $H \not\mapsto L$.

NDI_{sw} is strictly contained in NDI_{pw} : Containment is clear, since NDI_{pw} -security is NDI -security restricted to the case of singletons and thus follows directly from NDI .

The system in Example 1 separates NDI_{sw} and NDI_{pw} . For NDI_{pw} , it suffices to test if $H_1 \not\mapsto_I L$ because of symmetry. We have $H_1 \not\mapsto_I L$ because by visiting s_0 an appropriate number of times we can add any number of h_1 actions to a run without changing its L view. As already seen, this system is not NDI -secure; if L observes the view $0\ell 1$ the action sequence $\varepsilon \in \{h_1, h_2\}^*$ wasn’t performed by $\{H_1, H_2\}$. \square

7.2 Relationships between Cut-based Notions of GN

A *GN vulnerability* of a system \mathcal{M} is a tuple $(u, \alpha_0, a, \alpha_1, \beta, \mapsto)$, where \mapsto is a policy over the domain set of \mathcal{M} , u is a domain in \mathcal{M} , $\alpha_0, \alpha_1 \in A^*$, $a \in A$ with $\text{dom}(a) \not\mapsto u$ and $\beta \in \text{Views}_u(\mathcal{M})$ such that there is a run r that satisfies $\text{view}_u(r) = \beta$ and at least one of

- $\text{act}(r) = \alpha_0 \alpha_1$ and no run r' with $\text{act}(r') = \alpha_0 a \alpha_1$ satisfies $\text{view}_u(r') = \beta$,
- $\text{act}(r) = \alpha_0 a \alpha_1$ and no run r' with $\text{act}(r') = \alpha_0 \alpha_1$ satisfies $\text{view}_u(r') = \beta$.

If the context is clear, we only say *vulnerability*. We evidently have a vulnerability of a system if and only if $GN^+(\text{dom}(a), u)$ or $GN^-(\text{dom}(a), u)$ does not

hold. Without loss of generality we always assume a violation of $\text{GN}^+(\text{dom}(a), u)$ if there is a vulnerability, since the case of a $\text{GN}^-(\text{dom}(a), u)$ violation is similar.

The proofs to compare the different cut-based variants of GN are done by contraposition and show how, for given cuts \mathcal{C}_0 and \mathcal{C}_1 , a vulnerability of $\mathcal{M}^{\mathcal{C}_1}$ can be translated into a vulnerability of $\mathcal{M}^{\mathcal{C}_0}$. The next definition formalizes the idea that the view of an attacking coalition, e.g. the Low domain of a cut, has at least as much information as the view of a sub-coalition. We will need this to argue that if a coalition possesses enough information to successfully launch an attack on a system (i.e. it can violate GN^+) then, a fortiori, a bigger coalition possesses enough information for an attack.

Definition 6. Let \mathcal{M} be a system with action set A and observation set O , let \mathcal{D}_0 and \mathcal{D}_1 be abstractions of its domain set, and $F \in \mathcal{D}_0$, $G \in \mathcal{D}_1$ such that $F \subseteq G$. Then the operator

$$\text{pr}_F^G: A \cup O^G \cup \text{Views}_G(\mathcal{M}^{\mathcal{D}_1}) \rightarrow \text{Views}_F(\mathcal{M}^{\mathcal{D}_0})$$

is defined as follows:

- if $a \in A$ then $\text{pr}_F^G(a) = a|_F$, where a is considered to be a sequence of length one. The result is its subsequence of actions that F can perform, i.e. it is either a or ε ,
- if $o \in O^G$ then $\text{pr}_F^G(o) = o|_F$, that is observations made by G are restricted such that the result is the observation made by F ,
- if α is a G view and $\beta \in O^G \cup A_G \cdot O^G$ such that $\alpha\beta$ is a G view, then

$$\text{pr}_F^G(\alpha\beta) = \text{pr}_F^G(\alpha) \hat{\circ} \text{pr}_F^G(\beta).$$

For all other cases let the result be undefined. The symbol $\text{pr}_F^G(\cdot)$ is chosen to support the intuition that G views are ‘projected down’ to F views.

That the previous definition is reasonable is established by a correctness lemma which makes the restriction aspect of the operator clear.

Lemma 1. Let \mathcal{M} be a system, let \mathcal{D}_0 and \mathcal{D}_1 be abstractions of its domain set, and $F \in \mathcal{D}_0$, $G \in \mathcal{D}_1$ such that $F \subseteq G$. Then for all $r \in \text{Runs}(\mathcal{M})$ we have $\text{pr}_F^G(\text{view}_G(r)) = \text{view}_F(r)$.

Proof. First, we confine ourselves to case of observations. Let $o \in O^G$ such that $\text{obs}_G^{\mathcal{C}_1}(s) = o$ for some state s , then o is a function that maps each $u \in G$ to an element in O . The function $\text{pr}_F^G(o) = o|_F$ has the domain set F , and is total since o is total and we have $F \subseteq G$. We get that $\text{pr}_F^G(o)$ is the observation of domain F made in state s and is thus equal to $\text{obs}_F^{\mathcal{C}_0}(s)$.

The main result is shown by induction over runs. The base case follows from the previous paragraph. For the induction step, let r be a run of the form $r'as$, where $r \in \text{Runs}(\mathcal{M})$, a is an action and s a state of \mathcal{M} . We distinguish two cases.

If $\text{dom}^{C_1}(a) \neq G$ then we have $\text{view}_G(ras') = \text{view}_G(r) \hat{\circ} \text{obs}_G^{C_1}(s')$ and $\text{pr}_F^G(\text{view}_G(r) \hat{\circ} \text{obs}_G^{C_1}(s')) = \text{pr}_F^G(\text{view}_G(r)) \hat{\circ} \text{pr}_F^G(\text{obs}_G^{C_1}(s'))$, and this is equal to $\text{view}_F(r) \hat{\circ} \text{obs}_F^{C_0}(s')$ by induction and the special case of observations above. This value equals $\text{view}_F(ras')$ by definition of view. Note that we must have $\text{dom}^{C_0}(a) \neq F$, for if we had $\text{dom}^{C_0}(a) = F$ then $\text{dom}(a) \in F$, which implies $\text{dom}(a) \in G$ and yields $\text{dom}^{C_1}(a) = G$, contrary to the case assumption.

If $\text{dom}^{C_1}(a) = G$ then $\text{view}_G(ras') = \text{view}_G(r) \cdot a \cdot \text{obs}_G^{C_1}(s')$. Applying pr_F^G and induction yields $\text{pr}_F^G(\text{view}_G(ras')) = \text{view}_F(r) \hat{\circ} \text{pr}_F^G(a) \hat{\circ} \text{obs}_F^{C_0}(s')$, which is equal to $\text{view}_F(ras')$ in both cases $\text{dom}^{C_0}(a) = F$ and $\text{dom}^{C_0}(a) \neq F$. \square

Conditions under which the translation of a vulnerability is possible are established by the following result: the attacking coalition may not shrink and the translation must respect the status of being the attacker's victim.

Lemma 2. *Let \mathcal{M} be a system and \mapsto be a policy, and $\mathcal{C}_0 = (\mathcal{H}_0, \mathcal{L}_0)$ be a cut of the domain set of \mathcal{M} with respect to \mapsto . Let $(F, \alpha_0, a, \alpha_1, \beta, \mapsto^{C_0})$ be a vulnerability of \mathcal{M}^{C_0} . Let $\mathcal{C}_1 = (\mathcal{H}_1, \mathcal{L}_1)$ be a cut such that there is $G \in \{\mathcal{H}_1, \mathcal{L}_1\}$ with $\text{dom}^{C_1}(a) \not\mapsto^{C_1} G$ and $F \subseteq G$. Then there is $\beta' \in \text{Views}_G(\mathcal{M}^{C_1})$ such that a vulnerability of \mathcal{M}^{C_1} is given by $(G, \alpha_0, a, \alpha_1, \beta', \mapsto^{C_1})$.*

Proof. Since $(F, \alpha_0, a, \alpha_1, \beta, \mapsto^{C_0})$ is a vulnerability of \mathcal{M}^{C_0} , there is a run r on $\alpha_0\alpha_1$ and an F view of β such that no run on $\alpha_0a\alpha_1$ attains the F view β . Due to the prerequisites it suffices to show that there is a G view β' attained by some run on $\alpha_0\alpha_1$ such that no run on $\alpha_0a\alpha_1$ can attain a G view of β' , because then $(G, \alpha_0, a, \alpha_1, \beta', \mapsto^{C_1})$ is a vulnerability of \mathcal{M}^{C_1} , due to a violation of $\text{GN}^+(\text{dom}^{C_1}(a), G)$, and we are finished. By vulnerability, there is a run on $\alpha_0\alpha_1$ with F view of β ; let β' be G view of that run. If there were a run r on $\alpha_0a\alpha_1$ with $\text{view}_G(r) = \beta'$ then this run would satisfy $\text{view}_F(r) = \text{pr}_F^G(\text{view}_G(r)) = \text{pr}_F^G(\beta') = \beta$ by the same lemma, which contradicts the existence of the vulnerability of \mathcal{M}^{C_0} . Therefore, no such run can exist and we have found a vulnerability of \mathcal{M}^{C_1} as claimed. \square

Some relationships between cut-based variants of GN are trivial and can be seen directly.

Proposition 3. *C-GN implies both H-GN and L-GN.*

That High-up GN implies Cut GN, and therefore these notions are equivalent, might not be apparent, but can be explained by the fact that the Low component \mathcal{L} of a cut is the intersection of the Low components $\mathcal{L}_0, \dots, \mathcal{L}_{n-1}$ of n High-up cuts and thus can obtain no more information about High behaviour than each \mathcal{L}_i individually. If we can prevent each \mathcal{L}_i from obtaining any information about how High actions are interleaved into runs then the same must apply to \mathcal{L} as well.

Theorem 2. *The notions C-GN and H-GN are equivalent.*

Proof. Because of Proposition 3 it suffices to show that H-GN implies C-GN. The proof is done by contraposition and translates a vulnerability with respect to an arbitrary cut into a vulnerability with respect to a $\text{Hu}(\cdot)$ -style cut.

Let \mathcal{M} be a system with domain set D , \mapsto a policy over D and \mathcal{C}_0 a cut of D . Furthermore, let $(F, \alpha_0, a, \alpha_1, \beta, \mapsto^{\mathcal{C}_0})$ be a GN vulnerability of $\mathcal{M}^{\mathcal{C}_0}$. Set $\mathcal{C}_1 := \text{Hu}(\text{dom}(a))$, $\mathcal{H} := \text{dom}(a)^{\mapsto^{\mathcal{C}_1}}$ and $\mathcal{L} := D \setminus \text{dom}(a)^{\mapsto^{\mathcal{C}_1}}$. Then we have $\mathcal{C}_1 = (\mathcal{H}, \mathcal{L})$. We show that the prerequisites for Lemma 2 are satisfied, which gives us a vulnerability of $\mathcal{M}^{\mathcal{C}_1}$.

First, we demonstrate that $\text{dom}^{\mathcal{C}_1}(a) = \mathcal{H} \not\mapsto^{\mathcal{C}_1} \mathcal{L}$. Let $u \in \mathcal{H}$ and $v \in \mathcal{L}$, we must show that $u \not\mapsto v$. Assume $u \mapsto v$, then by choice of \mathcal{C}_1 we have $\text{dom}(a) \mapsto u$, which implies $\text{dom}(a) \mapsto u \mapsto v$ and $\text{dom}(a) \mapsto v$ by transitivity. Therefore $v \in \mathcal{H}$, which contradicts $v \in \mathcal{L}$, and hence we have $u \not\mapsto v$. It remains to prove that $F \subseteq \mathcal{L}$. Let $u \in F$, then due to vulnerability we have $\text{dom}^{\mathcal{C}_0}(a) \not\mapsto^{\mathcal{C}_0} F$, i.e. $\text{dom}(a) \not\mapsto u$. By choice of \mathcal{C}_1 we get $u \notin \mathcal{H}$, which is equivalent to $u \in \mathcal{L}$. Now application of Lemma 2 yields a vulnerability of $\mathcal{M}^{\mathcal{C}_1}$. \square

The result obtained by Theorem 2 shows completeness of Ryan’s technique for GN. From this follows that the High-up variant of GN implies the Low-down variant. There is also an example that demonstrates that these notions are distinct, and thus the High-up variant is stricter.

Theorem 3. *H-GN is strictly contained in L-GN.*

Proof. Containment follows from the facts that H-GN = C-GN by Theorem 2 and the trivial implications from Proposition 3. For separation, we recall Figure 6, and modify it slightly to suit our needs. This system can be verified to be GN-secure for the separation policy (i.e., the identity relation) on $\{H, L_1, L_2\}$; add the edges (L_1, H) and (L_2, H) to it and call it \mapsto . We anticipate the result that GN_{pw} is monotonic (see Proposition 7), and get that the system is GN_{pw} -secure for \mapsto .

With respect to \mapsto , the domain set has two Low-down cuts, which are $\text{Ld}(L_1)$ and $\text{Ld}(L_2)$. The systems $\mathcal{M}^{\text{Ld}(L_1)}$ and $\mathcal{M}^{\text{Ld}(L_2)}$ can be shown to be GN-secure for $\mapsto^{\text{Ld}(L_1)}$ and $\mapsto^{\text{Ld}(L_2)}$, respectively, and therefore \mathcal{M} is L-GN-secure for \mapsto . However, for the High-up cut $\text{Hu}(H)$, one can see that $\mathcal{M}^{\text{Hu}(H)}$ fails to be GN-secure for $\mapsto^{\text{Hu}(H)}$. Consider the run $r := s_0 h s'_0 \ell_1 s_5 \ell_2 s_{13}$. We have $\text{view}_L(r) = \begin{smallmatrix} \perp & 0 \\ \perp & \perp \end{smallmatrix} \begin{smallmatrix} \ell_1 & 0 \\ \ell_2 & \ell_1 \end{smallmatrix}$, where L observations are written in the form $\begin{smallmatrix} \text{obs}_{L_1}(\cdot) \\ \text{obs}_{L_2}(\cdot) \end{smallmatrix}$. By the parity of their final observations after performing r , domains L_1 and L_2 together can determine that H performed h at the very beginning of the run. Thus, $\mathcal{M}^{\text{Hu}(H)}$ doesn’t satisfy the property $\text{GN}^-(\{H\}, \{L_1, L_2\})$ for $\mapsto^{\text{Hu}(H)}$, which means that \mathcal{M} is not H-GN-secure for \mapsto . \square

The weakness of Low-down GN is that it assumes a somewhat restricted attacker that never groups domains into Low that may not interfere with each other according to the policy. (For example, for the policy in Figure 6, the coalition $\{L_1, L_2\}$ is not covered.) But nevertheless such coalitions are possible, which provides an argument against Low-down GN if coalitions are a risk. In a later subsection about monotonicity, we will show that Low-down GN is not

monotonic, which one can interpret as further evidence that it might seem problematic. However, Low-down GN doesn't break all our intuitions; as one might expect, it turns out to be stricter than GN_{pw} .

Theorem 4. *L-GN is strictly contained in GN_{pw} .*

Proof. Containment is shown by contraposition. Let \mathcal{M} be a system with domain set D and \mapsto a policy over D . Assume that \mathcal{M} is not GN_{pw} -secure for \mapsto and has a vulnerability $(u, \alpha_0, a, \alpha_1, \beta, \mapsto)$.

Set $\mathcal{C} := \text{Ld}(u)$, $\mathcal{L} := \mapsto u$ and $\mathcal{H} := D \setminus \mapsto u$. We show, using Lemma 1, that there is β' so that $(\mathcal{L}, \alpha_0, a, \alpha_1, \beta', \mapsto^{\mathcal{C}})$ is a vulnerability in $\mathcal{M}^{\mathcal{C}}$. First, we have $\text{dom}^{\mathcal{C}}(a) = \mathcal{H}$, due to $\text{dom}(a) \not\mapsto u$, which implies $\text{dom}^{\mathcal{C}}(a) \not\mapsto^{\mathcal{C}} \mathcal{L}$. Next, we demonstrate existence of a suitable β' . We identify observations made by v with observations made by the singleton coalition $\{v\}$, and consider the trivial abstraction of D , which is $\{\{w\} : w \in D\}$. Then we clearly have $\{v\} \subseteq \mathcal{L}$ and can apply Lemma 1. Due to vulnerability, there is a run on $\alpha_0 a \alpha_1$ which has a $\{u\}$ view of β such that no run on $\alpha_0 a \alpha_1$ has a $\{u\}$ view of β . Let β' be the \mathcal{L} view of this run. If there were a run r on $\alpha_0 a \alpha_1$ with \mathcal{L} view of β' , then $\text{view}_u(r) = \text{pr}_{\{u\}}^{\mathcal{L}}(\text{view}_{\mathcal{L}}(r)) = \text{pr}_{\{u\}}^{\mathcal{L}}(\beta') = \beta$ by identification of u and $\{u\}$ and Lemma 1, contradicting the violation of $\text{GN}^+(u, v)$ in \mathcal{M} . Therefore, no such run can exist and $(\mathcal{L}, \alpha_0, a, \alpha_1, \beta', \mapsto^{\mathcal{C}})$ is a vulnerability of $\mathcal{M}^{\mathcal{C}}$.

For separation, take the example from Theorem 3 and add the additional edge (L_1, L_2) to \mapsto . The system is still GN_{pw} -secure for \mapsto due to Proposition 7, but since we have $\{H\} \not\mapsto^{\text{Ld}(L_2)} \{L_1, L_2\}$, the system $\mathcal{M}^{\text{Ld}(L_2)}$ is not GN-secure by the argument in the proof of Theorem 3. \square

This concludes our study of cuts in the context of GN.

7.3 Relationships between Cut-based Notions of NDI

In this subsection, an *NDI vulnerability* of a system \mathcal{M} is a tuple $(u, \alpha, \beta, \mapsto)$, where \mapsto is a policy over the domain set of \mathcal{M} , u is a domain in \mathcal{M} , $\text{dom}(a) \not\mapsto u$ for all actions a that occur in α , and there is no run r of \mathcal{M} with $\text{act}_{\neq u}(r) = \alpha$ and $\text{view}_u(r) = \beta$. If the context is clear, we only say *vulnerability*. Clearly, a system is NDI_{sw} -secure if and only if it has no vulnerabilities.

We will follow the same strategy as used in the previous subsection, and first provide a lemma to translate vulnerabilities, then give proofs for the relationships claimed in Theorem 1.

In order to translate vulnerabilities from one cut to another, we again must make sure that the attacking coalition doesn't shrink. Additionally, since NDI_{sw} deals with combined behaviour, the translation must make sure that some non-interference constraints, which are pairs (u, v) such that $u \not\mapsto v$, are preserved.

Lemma 3. *Let \mathcal{M} be a system, \mapsto a policy over its domain set, $\mathcal{C}_0 = (\mathcal{H}_0, \mathcal{L}_0)$ be a cut of its domain set, and $F \in \{\mathcal{H}_0, \mathcal{L}_0\}$ such that $(F, \alpha, \beta, \mapsto^{\mathcal{C}_0})$ is a vulnerability of $\mathcal{M}^{\mathcal{C}_0}$. Let $\mathcal{C}_1 = (\mathcal{H}_1, \mathcal{L}_1)$ be a cut such that there is $G \in \{\mathcal{H}_1, \mathcal{L}_1\}$ with*

1. for all actions a that occur in α , we have $\text{dom}^{\mathcal{C}_1}(a) \not\mapsto^{\mathcal{C}_1} G$, and
2. $F \subseteq G$.

Then there is $\beta' \in \text{Views}_G(\mathcal{M}^{\mathcal{C}_1})$ so that $(G, \alpha, \beta', \mapsto^{\mathcal{C}_1})$ is a vulnerability of $\mathcal{M}^{\mathcal{C}_1}$.

Proof. Due to the prerequisites, it only remains to show the existence of a suitable β' . Let β' be a G view with $\text{pr}_F^G(\beta') = \beta$. Set $\mathcal{F} := \varphi_0 F$ and $\mathcal{G} := \varphi_1 G$. Prerequisite 1 gives us $\alpha \in A_G^*$. If there were a run r of $\mathcal{M}^{\mathcal{C}_1}$ with $\text{act}_{\mathcal{G}}(r) = \alpha$ and $\text{view}_G(r) = \beta'$, then the same run would satisfy $\text{act}_{\mathcal{F}}(r) = \alpha$, since by vulnerability α consists of actions by domains in \mathcal{F} only, and because we have $\text{view}_F(r) = \text{pr}_F^G(\text{view}_G(r)) = \text{pr}_F^G(\beta') = \beta$ by Lemma 1, which contradicts the vulnerability of $\mathcal{M}^{\mathcal{C}_1}$. Therefore, no such run can exist and $(G, \alpha, \beta', \mapsto^{\mathcal{C}_1})$ is a vulnerability in $\mathcal{M}^{\mathcal{C}_1}$. \square

Just as with GN, some relationships are trivial and can be seen from Definition 5 right away.

Proposition 4. *The notion C-NDI implies H-NDI and L-NDI.*

Contrary to GN, however, where High-up GN is strictly contained in Low-down GN, we have instead the somewhat surprising situation that the corresponding variants of NDI are incomparable. The next theorem provides the necessary examples.

Theorem 5. *The notions L-NDI and H-NDI are incomparable with respect to implication.*

Proof. *H-NDI does not imply L-NDI:* Consider the system and policy in Example 1. In the proof of Proposition 1 it is shown that the system violates NDI_{sw} -security with respect to the cut $\text{Ld}(L)$, which groups H_1 and H_2 together. It is therefore not L-NDI-secure. However, it is H-NDI-secure for the depicted policy. To show this, it is enough to prove NDI-security of the system with respect to $\text{Hu}(H_1)$, because the case $\text{Hu}(H_2)$ is symmetrical to it.

Set $\mathcal{C} := \text{Hu}(H_1)$, $\mathcal{H} := \{H_1\}$ and $\mathcal{L} := \{H_2, L\} \setminus H$. Then $\mathcal{C} = (\mathcal{H}, \mathcal{L})$ and $\mathcal{H} \not\mapsto^{\mathcal{C}} \mathcal{L}$. Let $\alpha \in \{h_1\}^*$ and β be an \mathcal{L} view. Then α has the form h_1^k for $k \geq 0$ and β is an element of the language described by one of the regular expressions $\frac{0}{\perp}(\frac{0}{\perp})^n(h_2\frac{0}{\perp})^m$ for $n, m \geq 0$, or $\frac{0}{\perp}(\frac{0}{\perp})^n(h_2\frac{0}{\perp})^m\ell\frac{1}{\perp}((l+h_2)\frac{1}{\perp})^k$ for $n \geq 0, m \geq 1$ and $k \geq 0$, where \mathcal{L} observations are noted as $\frac{\text{obs}_L(\cdot)}{\text{obs}_{H_2}(\cdot)}$. It is clear that there is a run that demonstrates the compatibility of α and β : the state s_1 can be visited k times for performing α . We therefore have $\mathcal{H} \not\mapsto_I \mathcal{L}$ and conclude that the system is H-NDI-secure.

L-NDI does not imply H-NDI: Consider the system in Figure 6. To prove it L-NDI-secure, it suffices to do so for the cut $\mathcal{C} := \text{Ld}(L_1)$, because the case of the only other $\text{Ld}(\cdot)$ cut is symmetric to it. Views perceived by $\{L_1\}$ (here, we identify $\{L_1\}$ with L_1) have the form $0(\ell_1 0)^n$ or $0(\ell_1 1)^n$ for $n \geq 0$. All these views are $\{L_1\}$ compatible with all $\alpha \in \{l_2, h\}^*$, because they can be attained by

the system performing l_1 only, or α can be added to a run by looping at states s_0, s_1, s_3, s_9 or s_{11} . The system is therefore L-NDI-secure.

However, it is not H-NDI-secure; take the cut $\text{Hu}(H)$, set $\mathcal{H} := \{H\}$ and $\mathcal{L} := \{L_1, L_2\}$, and consider the action sequence ε performed by domain \mathcal{H} . The \mathcal{L} view $\beta := \begin{smallmatrix} \perp & 0 & \perp \\ \perp & \ell_1 & \perp \\ \perp & \ell_2 & \perp \end{smallmatrix}$ can only be attained if the first action performed in the system is h . Therefore ε and β are not compatible and we have $\mathcal{H} \not\rightsquigarrow_I \mathcal{L}$. \square

These results show that Ryan’s technique is not ‘complete’ for Nondeducibility on Inputs, as High-up NDI and Low-down NDI are incomparable. The question if High-up NDI is complete can now be answered, because if High-up NDI implied Cut NDI, then High-up NDI would also imply Low-down NDI due to Proposition 4, which would contradict the result from Theorem 5. With symmetry, the same argument holds for High-up NDI and Low-down NDI swapped, and we have the following corollary.

Corollary 1. *C-NDI is strictly contained in both H-NDI and L-NDI.*

The previous theorem alone doesn’t yield evidence in favor of High-up or Low-down. As with GN, one might expect the High-up variant to be more adequate, but it turns out that this isn’t the case. We can argue against High-up using the system presented in Example 1. As shown in the proof of Theorem 5, it is High-up NDI- but not Low-down NDI-secure due to the cut $\text{Ld}(L)$ introducing a vulnerability. The cut $\text{Ld}(L)$ aggregates the domains into $\{L\}$ and $\not\rightsquigarrow L = \{H_1, H_2\}$. But if L can infer from observing a certain view that the domains in $\not\rightsquigarrow L$ did not perform some action sequence, this means that the system is not NDI_{sw} -secure. In other words, NDI_{sw} and Low-down NDI are equivalent notions on the example.

Corollary 2. *H-NDI does not imply NDI_{sw} .*

The weakness of High-up NDI is, similar to Low-down GN, that it doesn’t group domains into High that are incomparable in the policy, while NDI does. The definition of NDI is a natural extension of the pointwise application of two-level Nondeducibility on Inputs, so there is an argument that High-up is not adequate in the setting of NDI. The case against it can be made even stronger by proving that Low-down NDI does not have this undesirable property, which is what the next result accomplishes.

Theorem 6. *L-NDI implies NDI_{sw} .*

Proof. Let \mathcal{M} be a system with domain set D and \mapsto be a policy over D . Assume \mathcal{M} is not NDI_{sw} -secure for \mapsto , then there are $u \in D$, $\alpha \in A_{\not\rightsquigarrow u}^*$ and $\beta \in \text{Views}_u(\mathcal{M})$ so that $(u, \alpha, \beta, \mapsto)$ is a vulnerability of \mathcal{M} . Clearly, for the abstraction \mathcal{D} given by $\{\{v\} : v \in D\}$ the system $\mathcal{M}^{\mathcal{D}}$ has the vulnerability $(\{u\}, \alpha, \beta, \mapsto^{\mathcal{D}})$.

Consider the cut $\mathcal{C} := \text{Ld}(u)$, we use Lemma 3 to show that there is a vulnerability of $\mathcal{M}^{\mathcal{C}}$. Set $\mathcal{L} := \mapsto u$ and $\mathcal{H} := D \setminus \mapsto u$. We obviously have $\{u\} \subseteq \mathcal{L}$, so it only remains to show that α consists only of actions performed by domains

that may not interfere with \mathcal{L} with respect to $\mapsto^{\mathcal{C}}$. Let a be an action that occurs in α . Since $\text{dom}(a) \not\vdash u$ by vulnerability in $\mathcal{M}^{\mathcal{D}}$ we get that $\text{dom}(a) \notin \mathcal{L}$ by choice of \mathcal{C} , which means $\text{dom}(a) \in \mathcal{H}$, and this implies $\text{dom}^{\mathcal{C}}(a) = \mathcal{H}$ and therefore $\text{dom}^{\mathcal{C}}(a) \not\vdash^{\mathcal{C}} \mathcal{L}$.

Application of Lemma 3 now gives us a vulnerability of $\mathcal{M}^{\mathcal{C}}$, which means that \mathcal{M} is not L-NDI-secure for \mapsto . \square

For the last relationship, the notion High-up NDI is compared with NDI_{pw} . Recall that NDI_{pw} doesn't properly deal with combined behaviour, but this is what one would expect from a sensible Nondeducibility notion for multi-domain policies. The statement made by the next proposition therefore shouldn't be interpreted as redeeming.

Proposition 5. *H-NDI implies NDI_{pw} .*

Proof. Let \mathcal{M} be a system with domain set D and \mapsto a policy over D . Assume that \mathcal{M} is not NDI_{pw} -secure for \mapsto , then there are $u, v \in D$ with $u \not\vdash v$ and $u \rightsquigarrow_I v$. We can proceed as in the proof of Theorem 6: the system $\mathcal{M}^{\mathcal{D}}$ has a vulnerability $(\{v\}, \alpha, \beta, \mapsto^{\mathcal{D}})$, where $\mathcal{D} := \{\{w\} : w \in D\}$, and choosing $\mathcal{C} := \text{Hu}(u)$ yields a vulnerability of $\mathcal{M}^{\mathcal{C}}$ via Lemma 3.

(An important point here is that α consists of actions by a single domain only, whereas in the proof of Theorem 6 the sequence α can contain actions by multiple domains. If only a single domain u is acting, a High-up cut can capture u in its abstracted High domain; in the case of multiple active domains it might not, see Example 1.) \square

7.4 Relationships between the GN and NDI Variants

Proposition 6. *The notion GN_{pw} doesn't imply any of H-NDI, L-NDI or C-NDI. The notion L-GN does not imply H-NDI or C-NDI.*

Proof. The system depicted in Figure 6 is GN_{pw} - but not H-GN-secure as argued in the proof of Theorem 3 for the corresponding policy and therefore not H-NDI-secure (since H-GN implies H-NDI). As a trivial consequence, we get that GN_{pw} doesn't imply C-NDI. Also, we have that L-GN doesn't imply H-NDI, since otherwise H-NDI implied L-GN and, since L-GN implies L-NDI, also L-NDI, contradicting Theorem 5. As a consequence, we find that L-GN doesn't imply C-NDI.

To see that GN_{pw} doesn't imply L-NDI, add the edge (L_1, L_2) to the policy considered in the previous paragraph, and consider the cut $\text{Ld}(L_2)$, which is given by $(\{H\}, \{L_1, L_2\})$ and is the same cut used in the proof of Theorem 3 to demonstrate a violation of H-NDI-security. Therefore, the system is not L-NDI-secure either. \square

If all results from this subsection are combined, we obtain exactly the containment diagram as claimed by Theorem 1.

7.5 Monotonicity

The statement $u \not\vdash v$ can be understood as a noninterference constraint and adding the edge $u \mapsto v$ removes this constraint from a policy. If a system is secure (for a sensible definition of ‘secure’) and constraints are discarded from the policy, it seems reasonable to expect that security is preserved. In this subsection we investigate which of our notions support this intuition.

We have to compare cuts of the same domain set but with respect to different policies, which is why we make explicit which policy a cut refers to by writing, for example, $\text{Hu}^{\mapsto}(\cdot)$.

If not mentioned otherwise, all systems in this subsection refer to their set of domains as D , and we have two policies \mapsto_0 and \mapsto_1 with $\mapsto_0 \subseteq \mapsto_1$.

Theorem 7. *The notions GN, H-GN and C-GN are monotonic.*

Proof. GN is monotonic: Let $(u, \alpha_0, a, \alpha_1, \beta, \mapsto_1)$ be a GN-vulnerability of some system \mathcal{M} , then $\text{dom}(a) \not\vdash_1 u$, which implies $\text{dom}(a) \not\vdash_0 u$ since $\not\vdash_1 \subseteq \not\vdash_0$. Because domain assignments and u views are not affected by the policy, we have that $(u, \alpha_0, a, \alpha_1, \beta, \mapsto_0)$ is a GN vulnerability of \mathcal{M} .

H-GN and C-GN are monotonic: Since H-GN and C-GN are equivalent by Theorem 2, it suffices to prove it for H-GN only.

Set $\mathcal{C}_1 := \text{Hu}^{\mapsto_1}(\text{dom}(a))$, $\mathcal{H}_1 := \text{dom}(a)^{\mapsto_1}$ and $\mathcal{L}_1 := D \setminus \text{dom}(a)^{\mapsto_1}$. Let \mathcal{M} be a system such that $(\mathcal{L}_1, \alpha_0, a, \alpha_1, \beta, \mapsto_1)$ is a GN vulnerability of $\mathcal{M}^{\mathcal{C}}$. Then we have $\mathcal{H}_1 \not\vdash_1^{\mathcal{C}_1} \mathcal{L}_1$. Furthermore, set $\mathcal{C}_0 := \text{Hu}^{\mapsto_0}(\text{dom}(a))$, $\mathcal{H}_0 := \text{dom}(a)^{\mapsto_0}$, and $\mathcal{L}_0 := D \setminus \text{dom}(a)^{\mapsto_0}$. This implies $\mathcal{H}_0 \not\vdash_0^{\mathcal{C}_0} \mathcal{L}_0$. We show that $(\mathcal{L}_0, \alpha_0, a, \alpha_1, \mapsto_0)$ is a GN vulnerability of $\mathcal{M}^{\mathcal{C}_0}$ by demonstrating that we have $\mathcal{L}_1 \subseteq \mathcal{L}_0$ and $\text{dom}^{\mathcal{C}_0}(a) \not\vdash_0^{\mathcal{C}_0} \mathcal{L}_0$, and then applying Lemma 2.

For the former, let $u \in \mathcal{L}_1$. Then $\text{dom}(a) \not\vdash_1 u$ which implies $\text{dom}(a) \not\vdash_0 u$, and this gives us $u \in \mathcal{L}_0$. For the latter, by choice of \mathcal{C}_0 we clearly have that $\text{dom}(a) \in \mathcal{H}_0$ and therefore $\text{dom}^{\mathcal{C}_0}(a) = \mathcal{H}_0$, which implies $\text{dom}^{\mathcal{C}_0}(a) \not\vdash_0^{\mathcal{C}_0} \mathcal{L}_0$. Lemma 2 now gives us the existence of a GN vulnerability of $\mathcal{M}^{\mathcal{C}_0}$. \square

That High-up GN is monotonic can be explained with the fact that adding edges can never grow the Low component in a given High-up cut. Since according to the definition, High is taken to be u^{\mapsto} for a given domain u , adding an edge might grow High, which in turn would shrink Low. So adding edges can never increase the knowledge of the Low coalition when High-up cuts are used.

The case is different for Low-down GN. Adding edges to a policy can join two formerly incomparable elements which then become members of Low in a Low-down coalition. The reason is the definition of the Low component, which for a given domain u is taken to be $\mapsto u$, or in other words, Low will be all domains from which u is permitted to learn. Protection due to Low-down cuts thus requires a somewhat friendly attacker, so there is an argument that this can be considered a flaw in the definition itself.

Proposition 7. *L-GN is not monotonic.*

Proof. As shown in the proof of Theorem 3, the system depicted in Figure 6 is L-GN-secure. The proof of Theorem 4 demonstrates that adding the edge (L_1, L_2) to the policy turns it non-L-GN-secure. \square

This concludes the investigation of monotonicity for our variants of GN. But there are two more things to note about adding edges to a policy when GN is used: (1) The attack surface does not increase, since already generic GN requires all domains to be unable to infer the occurrence or nonoccurrence of *any* secret events. In fact, it might even get smaller. (2) Adding edges can decrease the number of cuts, so there might be fewer requirements on the system in order to be secure. For Cut GN, the results from Theorem 7 suggests that a smaller number of cuts can compensate for a possible increase of Low's knowledge.

The argument for Cut NDI being monotonic is similar to the case of High-up and Cut GN. That NDI and NDI_{pw} are monotonic is straightforward.

Proposition 8. *The notions NDI_{pw} , NDI and C-NDI are monotonic.*

Proof. NDI_{pw} is monotonic: Let \mathcal{M} be NDI_{pw} -secure for \mapsto_0 and $u, v \in D$ such that $u \rightsquigarrow_I v$. Then $u \mapsto_0 v$ by NDI_{pw} -security of \mathcal{M} for \mapsto_0 and $u \mapsto_1 v$ by $\mapsto_0 \subseteq \mapsto_1$. Therefore, \mathcal{M} is NDI_{pw} -secure for \mapsto_1 .

C-NDI is monotonic: Let \mathcal{M} be a system and $\mathcal{C} = (\mathcal{H}, \mathcal{L})$ be a cut of D , then we have $H \not\mapsto_1^{\mathcal{C}} L$. Assume $\mathcal{M}^{\mathcal{C}}$ has an NDI vulnerability $(\mathcal{L}, \alpha, \beta, \mapsto_1)$. We show that $(\mathcal{L}, \alpha, \beta, \mapsto_0)$ is a vulnerability of that system, too.

First, let $u \in \mathcal{H}$ and $v \in \mathcal{L}$. Then we have $u \not\mapsto_1 v$, which implies $u \not\mapsto_0 v$. Therefore $\mathcal{H} \not\mapsto_0^{\mathcal{C}} \mathcal{L}$ and \mathcal{C} is a valid cut with respect to \mapsto_0 . This also gives us $\text{dom}^{\mathcal{C}}(a) \not\mapsto_0^{\mathcal{C}} \mathcal{L}$ for all actions a that occur in α , because we must have $\text{dom}^{\mathcal{C}}(a) = \mathcal{H}$ by vulnerability.

Finally, clearly β remains a valid \mathcal{L} view and trivially we have $\mathcal{L} \subseteq \mathcal{L}$. Therefore, due to Lemma 3, we get that $(\mathcal{L}, \alpha, \beta, \mapsto_0)$ is an NDI vulnerability of $\mathcal{M}^{\mathcal{C}}$.

NDI is monotonic: Let \mathcal{M} be a system that is NDI-secure for \mapsto_0 , $u \in D$, $X_0 := \not\mapsto_0 u$, $X_1 := \not\mapsto_1 u$, $\alpha \in A_{X_1}^*$ and $\beta \in \text{Views}_u(\mathcal{M})$. Then, because of $\not\mapsto_1 \subseteq \not\mapsto_0$, we have $X_1 \subseteq X_0$, which implies $\alpha \in A_{X_0}^*$. With NDI-security of \mathcal{M} for \mapsto_0 we get that α and β are compatible, and therefore \mathcal{M} is NDI-secure for \mapsto_1 as well. \square

Merging two incomparable domains into a Low-down cut is also possible for NDI, and here we too obtain the result that the Low-down variant fails to be monotonic. In fact, the same system as in the case of Low-down GN can be used.

However, the notion High-up NDI does not share the monotonicity property with its GN counterpart. While it's true that Low might shrink if edges are added and thus has less knowledge at hand for an attack, and that the new policy might have fewer cuts, the set of action sequences that have to be compatible with any Low view grows, which increases the attack surface (i.e., Low might now be able to exclude certain High behaviours). The next result suggests that this increase can outweigh the loss of knowledge experienced by Low and the fewer number of cuts combined.

Theorem 8. *The notions L-NDI and H-NDI are not monotonic.*

Proof. L-NDI is not monotonic: In the proof of Theorem 5 it is shown that the system in Figure 6 is L-NDI-secure for the depicted policy, which we call \mapsto_0 , given by $L_1 \mapsto_0 H$ and $L_2 \mapsto_0 H$ (excluding edges due to reflexivity). Let \mapsto_1 be the policy obtained by taking \mapsto_0 and adding the additional edge $L_1 \mapsto_1 L_2$. Then the system is not L-NDI-secure for \mapsto_1 : consider the cut $\text{Ld}^{\mapsto_1}(L_2)$. It is equivalent to $\text{Hu}^{\mapsto_0}(H)$. In the proof of Theorem 5, it is demonstrated that the system is not NDI-secure for $\mapsto_0^{\text{Hu}(H)}$. But since $\text{Hu}(H)$ and $\text{Ld}(L_2)$ are equal, we get that it isn't NDI-secure for $\mapsto_1^{\text{Ld}(L_2)}$ either.

H-NDI is not monotonic: Consider the system \mathcal{M} in Example 1 and call the depicted policy \mapsto_0 , which is given by $L \mapsto_0 H_1$ and $L \mapsto_0 H_2$, excluding edges due to reflexivity. Let \mapsto_1 be the policy \mapsto_0 with the additional edge $H_1 \mapsto_1 H_2$. The system is H-NDI-secure for \mapsto_0 due to Theorem 5, but it is not H-NDI-secure for \mapsto_1 . To see this, take the cut $\text{Hu}^{\mapsto_1}(H_1)$, which is equivalent to $\text{Ld}^{\mapsto_0}(L)$. And as argued in the proof of Proposition 1, the system $\mathcal{M}^{\text{Ld}^{\mapsto_0}(L)}$ has an NDI vulnerability, and thus \mathcal{M} is not H-NDI-secure for \mapsto_1 . \square

Since NDI is monotonic, it cannot be equal to Low-down NDI, and therefore this containment must be strict. With the same argument we get strict containment of High-up NDI in pointwise NDI.

Corollary 3. *L-NDI is strictly contained in NDI, and H-NDI is strictly contained in NDI_{pw} .*

Together with the results on containment relationships obtained in the previous subsections, our results on monotonicity now yield proofs for Theorem 1.

8 Conclusion

In this work we have discussed several variants of Generalized Noninterference and Nondeducibility on Inputs for multi-domain policies that use reductions to the two-level case, including a technique proposed by Ryan. We have found that this technique leads to a stricter notion in the case of Generalized Noninterference, but behaves counter-intuitively in the case of Nondeducibility on Inputs, where it yields a notion that is incomparable to a natural variant for multi-domain policies. We have found evidence that seems to suggest that considering all cuts is a more robust choice as a reduction technique. Some notions we obtained break our intuitions in the sense that they are not preserved under removing noninterference constraints.

These results have left open a question about how to handle the general case of collusion, as reductions to $H \not\mapsto L$ are a special case of collusion where two coalitions are operating, while general abstractions can model an arbitrary number of coalitions. It seems natural to extend the theory such that it can handle general abstractions, but then we leave the area of transitive noninterference. For example, consider the transitive policy \mapsto that contains the relations $A \mapsto B$ and $C \mapsto D$ only, and the abstraction \mathcal{D} that forms the coalitions $\{A\}$, $\{B, C\}$ and

$\{D\}$. The resulting policy $\mapsto^{\mathcal{D}}$ is intransitive, as it has edges $\{A\} \mapsto^{\mathcal{D}} \{B, C\}$ and $\{B, C\} \mapsto^{\mathcal{D}} \{D\}$, but lacks the edge $\{A\} \mapsto^{\mathcal{D}} \{D\}$. In this case, it seems reasonable to say that information may get from A to D , as domains B and C collude and share their observations, but it needs intermediate behaviour by them in order to forward the information. Adding the edge $\{A\} \mapsto^{\mathcal{D}} \{D\}$ clashes with this reasoning, as it would express that A may *directly* communicate with D . This suggests that dealing with general abstractions requires techniques from the theory of intransitive noninterference. Semantics for intransitive noninterference that build in types of collusion have been considered in a few works [22,21], but the relationship of these definitions to abstractions remains to be studied.

References

1. Haigh, J.T., Young, W.D.: Extending the noninterference version of MLS for SAT. *IEEE Transactions on Software Engineering* **13**(2) (1987) 141
2. Rushby, J.: Noninterference, transitivity, and channel-control security policies. Technical report, SRI international (Dec 1992)
3. van der Meyden, R.: What, indeed, is intransitive noninterference? *Journal of Computer Security* **23**(2) (2015) 197–228 (Extended version of a paper in ESORICS’2007.).
4. Goguen, J.A., Meseguer, J.: Security Policies and Security Models. In: 1982 IEEE Symposium on Security and Privacy, Oakland, CA, USA, April 26–28, 1982. (1982) 11–20
5. Sutherland, D.: A model of information. In: Proc. 9th National Computer Security Conference, DTIC Document (1986) 175–183
6. McCullough, D.: Foundations of Ulysses: The theory of security. Technical report, DTIC Document (1988)
7. McLean, J.: A general theory of composition for trace sets closed under selective interleaving functions. In: Research in Security and Privacy, 1994. Proceedings., 1994 IEEE Computer Society Symposium on, IEEE (1994) 79–93
8. Mantel, H.: Possibilistic definitions of security - an assembly kit. In: Computer Security Foundations Workshop, 2000. CSFW-13. Proceedings. 13th IEEE, IEEE (2000) 185–199
9. Focardi, R., Gorrieri, R.: Classification of security properties. In: FOSAD 2000, LNCS 2171. (2001) 331–396
10. Roscoe, A.W.: CSP and determinism in security modelling. In: Proc. IEEE Symposium on Security and Privacy. (1995) 114–221
11. Ryan, P.Y.: Mathematical models of computer security. In: Foundations of Security Analysis and Design. Springer (2000) 1–62
12. Forster, R.: Non-interference properties for nondeterministic processes. PhD thesis, Dissertation for transfer to D.Phil status, Oxford University Computing Laboratory (1997)
13. Mantel, H.: A uniform framework for the formal specification and verification of information flow security. PhD thesis, Universität des Saarlandes (2003)
14. Millen, J.K.: Unwinding forward correctability. In: Proc. IEEE Computer Security Foundations Workshop. (1994) 2–10
15. Roscoe, A.W., Woodcock, J., Wulf, L.: Non-interference through determinism. *Journal of Computer Security* **4**(1) (1996) 27–54

16. Sutherland, D.: A model of information. In: Proc. National Computer Security Conference. (1986) 175–183
17. McCullough, D.: Noninterference and the composability of security properties. In: Proceedings of the 1988 IEEE Symposium on Security and Privacy, Oakland, California, USA, April 18-21, 1988. (1988) 177–186
18. Eggert, S., van der Meyden, R.: Dynamic Intransitive Noninterference Revisited. CoRR (2016) arXiv:1601.05187 [cs.CR].
19. Goguen, J.A., Meseguer, J.: Unwinding and Inference Control. In: Proceedings of the 1984 IEEE Symposium on Security and Privacy, Oakland, California, USA, April 29 - May 2, 1984. (1984) 75–87
20. van der Meyden, R., Zhang, C.: Algorithmic Verification of Noninterference Properties. *Electr. Notes Theor. Comput. Sci.* **168** (2007) 61–75
21. Engelhardt, K., van der Meyden, R., Zhang, C.: Intransitive Noninterference in Nondeterministic Systems. In: Proceedings of the 2012 ACM conference on Computer and communications security, ACM (2012) 869–880
22. Backes, M., Pfitzmann, B.: Intransitive non-interference for cryptographic purposes. In: IEEE Symposium on Security and Privacy. (2003) 140–152